

Вспомним про хеш

Хеш-функция — функция, осуществляющая преобразование массива входных данных произвольной длины в выходную битовую строку установленной длины, выполняемое определенным алгоритмом.

Преобразование, производимое хеш-функцией, называется хешированием. Результат преобразования называется хешем.

Хеш-функции применяются в следующих случаях:

При построении ассоциативных массивов.

При поиске дубликатов в последовательностях наборов данных.

При построении уникальных идентификаторов для наборов данных.

При вычислении контрольных сумм от данных для последующего обнаружения в них ошибок, возникающих при хранении и передаче данных.

При сохранении паролей в системах защиты в виде хеш-кода (для восстановления пароля по хеш-коду требуется функция, являющаяся обратной по отношению к использованной хеш-функции).

При выработке электронной подписи (на практике часто подписывается не само сообщение, а его хеш-образ).

MD5 — алгоритм хеширования, разработанный профессором Рональдом Л. Ривестом из Массачусетского технологического института в 1991 году. Предназначен для создания контрольных сумм или «отпечатков» сообщения произвольной длины и последующей проверки их подлинности. Алгоритм MD5 основан на алгоритме MD4.

Как работает протокол?

Утилита md5sum, предназначенная для хеширования данных заданного файла по алгоритму MD5, возвращает строку. Она состоит из 32 цифр в шестнадцатеричной системе счисления (016f8e458c8f89ef75fa7a78265a0025).

То есть хеш, полученный от функции, работа которой основана на этом алгоритме, выдает строку в 16 байт (128) бит. И эта строка включает в себя 16 шестнадцатеричных чисел. При этом изменение хотя бы одного ее символа приведет к последующему бесповоротному изменению значений всех остальных битов строки.

В данном алгоритме предполагается наличие 5 шагов, а именно:

Выравнивание потока

Добавление длины сообщения

Инициализация буфера

Вычисление в цикле

Результат вычислений

На первом шаге “Выравнивание потока” сначала дописывают единичный бит в конец потока, затем необходимое число нулевых бит. Входные данные выравниваются так, чтобы их новый размер был сравним с 448 по модулю 512. Выравнивание происходит, даже если длина уже сравнима с 448.

На втором шаге в оставшиеся 64 бита дописывают 64-битное представление длины данных до выравнивания. Сначала записывают младшие 4 байта. Если длина превосходит $2^{64}-1$, то дописывают только младшие биты. После этого длина потока станет кратной 512. Вычисления будут основываться на представлении этого потока данных в виде массива слов по 512 бит.

На третьем для вычислений используются четыре переменные размером 32 бита и задаются начальные значения в 16-ричном виде. В этих переменных будут храниться результаты промежуточных вычислений.

Во время 4-го шага “Вычисление в цикле” происходит 4 раунда, в которых сохраняются значения, оставшиеся после операций над предыдущими блоками. После всех операций суммируются результаты двух последних циклов. Раундов в MD5 стало 4 вместо 3 в MD4. Добавилась новая константа для того, чтобы свести к минимуму влияние входного сообщения. В каждом раунде на каждом шаге и каждый раз константа разная. Она суммируется с результатом и блоком данных. Результат каждого шага складывается с результатом предыдущего шага. Из-за этого происходит более быстрое изменение результата. Изменился порядок работы с входными словами в раундах 2 и 3.

В итоге на 5-ом шаге мы получим результат вычислений, который находится в буфере -это и есть хеш. Если выводить побайтово, начиная с младшего байта первой переменной и закончив старшим байтом последней, то мы получим MD5-хеш.

Уязвимости MD5

Алгоритм MD5 уязвим к некоторым атакам. Например, возможно создание двух сообщений с одинаковой хеш-суммой.

На данный момент существуют несколько видов взлома хешей MD5 — подбора сообщения с заданным хешем:

Перебор по словарю

Brute-force

RainbowCrack

Коллизия хеш-функции

При этом методы перебора по словарю и brute-force могут использоваться для взлома хеша других хеш-функций (с небольшими изменениями алгоритма). RainbowCrack требует предварительной подготовки радужных таблиц, которые создаются для заранее определённой хеш-функции. Поиск коллизий специфичен для каждого алгоритма. Рассмотрим каждый вид «взлома» по отдельности.

Атаки переборного типа

В криптографии атака полного перебора или исчерпывающий поиск ключей -это стратегия, которая теоретически может быть использована против любых зашифрованных данных.

Злоумышленник, который не может воспользоваться слабостью в системе шифрования, реализовывает атаку подобного типа. Она включает в себя систематическую проверку всех возможных ключей, пока не будет найден правильный. В худшем случае для взлома сообщения потребуется задействовать всю вычислительную мощность.

Перебор по словарю — атака на систему защиты, применяющая метод полного перебора предполагаемых паролей, используемых для аутентификации, осуществляемого путём последовательного пересмотра всех слов (паролей в чистом виде) определённого вида и длины из словаря с целью последующего взлома системы и получения доступа к секретной информации.

Как видно из определения, атаки по словарю являются атаками полного перебора. Единственное отличие состоит в том, что данные атаки обычно более эффективны так как становится не нужным перебирать все комбинации символов, чтобы добиться успеха.

Злоумышленники используют обширные списки наиболее часто используемых паролей таких как, имена домашних животных, вымышленных персонажей или конкретно характерных слов из словаря – отсюда и название атаки. Однако если пароль действительно уникален (не является комбинацией слов), атака по словарю не сработает. В этом случае использование атаки полного перебора -единственный вариант.

Для полного перебора или перебора по словарю можно использовать программы PasswordsPro, MD5BFCPF, John the Ripper. Для перебора по словарю существуют готовые словари.

RainbowCrack

Это ещё один метод взлома хеша. Он основан на генерировании большого количества хешей из набора символов, чтобы по получившейся базе вести поиск заданного хеша.

Радужные таблицы состоят из хеш-цепочек и более эффективны, чем предыдущий упомянутый тип атак, поскольку они оптимизируют требования к хранению, хотя поиск выполняется немного медленнее. Радужные таблицы отличаются от хеш-таблиц тем, что они создаются с использованием как хеш-функций, так и функций редукции.

Цепочки хешей — метод для уменьшения требования к объёму памяти. Главная идея — определение функции редукции R , которая сопоставляет значениям хеша значения из таблицы. Стоит отметить, что R не является обращением хеш-функции.

Радужные таблицы являются развитием идеи таблицы хеш-цепочек. Функции редукции применяются по очереди, перемежаясь с функцией хеширования.

Использование последовательностей функций редукции изменяет способ поиска по таблице. Поскольку хеш может быть найден в любом месте цепочки, необходимо сгенерировать несколько различных цепочек.

Существует множество систем взлома паролей и веб-сайтов, которые используют подобные таблицы. Основная идея данного метода — достижение компромисса между временем поиска по таблице и занимаемой памятью. Конечно, использование радужных таблиц не гарантирует 100% успеха взлома систем паролей. Но чем больше набор символов, используемый для создания радужной таблицы, и чем продолжительнее хеш-цепочки, тем больше будет шансов получить доступ к базе данных исходных паролей.

Коллизии MD5

Коллизия хеш-функции — это получение одинакового значения функции для разных сообщений и идентичного начального буфера. В отличие от коллизий, псевдоколлизии определяются как равные значения хеша для разных значений начального буфера, причём сами сообщения могут совпадать или отличаться. В 1996 году Ганс Доббертин нашёл псевдоколлизии в MD5, используя определённые инициализирующие векторы, отличные от стандартных. Оказалось, что можно для известного сообщения построить второе такое, что оно будет иметь такой же хеш, как и исходное. С точки зрения математики, это означает следующее:

$$\text{MD5}(I_V, L_1) = \text{MD5}(I_V, L_2),$$

где I_V — начальное значение буфера, а L_1 и L_2 — различные сообщения.

MD5 был тщательно изучен криптографическим сообществом с момента его первоначального выпуска и до 2004 года демонстрировал лишь незначительные недостатки. Однако летом 2004 года криптографы Ван Сяюнь и Фэн Дэngo продемонстрировали алгоритм способный генерировать MD5-коллизии с использованием стандартного вектора инициализации.

Позже данный алгоритм был усовершенствован, как следствие время поиска пары сообщений значительно уменьшилось, что позволило находить коллизии с приемлемой вычислительной сложностью. Как оказалось, в MD5 вопрос коллизий не решается.

Безопасное использование MD5

MD5 – до сих пор является одним из самых распространенных способов защитить информацию в сфере прикладных исследований, а также в области разработки веб-приложений. Хеш необходимо обезопасить от всевозможных хакерских атак.

Информационная энтропия

Надежность и сложность пароля в сфере информационных технологий обычно измеряется в терминах теории информации. Чем выше информационная энтропия, тем надежнее пароль и, следовательно, тем труднее его взломать.

Чем длиннее пароль и чем больше набор символов, из которого он получен, тем он надежнее. Правда вместо количества попыток, которые необходимо предпринять для угадывания пароля, принято вычислять логарифм по основанию 2 от этого числа, и полученное значение называется количеством «битов энтропии» в пароле. При увеличении длины пароля на один бит количество возможных паролей удвоится, что сделает задачу атакующего в два раза сложнее. В среднем, атакующий должен будет проверить половину из всех возможных паролей до того, как найдет правильный. В качестве наилучшей практики должно выполняться предварительное требование: приложение настаивает на том, чтобы пользователь использовал надежный пароль в процессе регистрации.

Добавление “соли” к паролю
Одна из наиболее распространенных причин успешных атак заключается в том, что компании не используют добавление соли к исходному паролю. Это значительно облегчает хакерам взлом системы с помощью атак типа радужных таблиц, особенно учитывая тот факт, что многие пользователи используют очень распространенные, простые пароли, имеющие одинаковые хеши.

Соль-это вторичный фрагмент информации, состоящий из строки символов, которые добавляются к открытому тексту (исходному паролю пользователя), а затем хешируется. Соление делает пароли более устойчивыми к атакам типа радужных таблиц, так как подобный пароль будет иметь более высокую информационную энтропию и, следовательно, менее вероятное существование в предварительно вычисленных радужных таблицах. Как правило, соль должна быть не менее 48 бит.

Декодирование кода MD5

Иногда при работе с компьютером или поврежденными базами данных требуется декодировать зашифрованное с помощью MD5 значение хеша.

Удобнее всего использовать специализированные ресурсы, предоставляющие возможность сделать это online:

md5.web-max.ca -данный сервис обладает простым и понятным интерфейсом. Для получения декодированного значения нужно ввести хеш и заполнить поле проверочной капчи;

md5decrypter.com -аналогичный сервис;

msurf.ru -данный ресурс имеет простой русскоязычный интерфейс. Его функционал позволяет не только расшифровывать значения хеш-кодов, но и создавать их.

Если присмотреться к значениям декодинга, то становится понятно, что процесс расшифровки почти не дает результатов. Эти ресурсы представляют собой одну или несколько объединенных между собой баз данных, в которые занесены расшифровки самых простых слов.

При этом данные декодирования хеша MD5 даже такой распространенной части пароля, как «админ», нашлись лишь в одной базе. Поэтому хеши паролей, состоящих из более сложных и длинных комбинаций символов, практически невозможно расшифровать.

Создание хеша MD5 является односторонним процессом. Поэтому не подразумевает обратного декодирования первоначального значения.

Заключение

Как уже отмечалось ранее, основная задача любой функции хеширования сообщений -производить образы, которые можно считать относительно случайными. Чтобы считаться криптографически безопасной, хэш-функция должна отвечать двум основным требованиям. Во-первых, злоумышленник не может сгенерировать сообщение, соответствующее определенному хеш-значению. Во-вторых, невозможно создать два сообщения, которые производят одно и то же значение (коллизии в MD5).

К сожалению, выяснилось, что алгоритм MD5 не способен отвечать данным требованиям. IETF (Internet Engineering Task Force) рекомендовала новым проектам протоколов не использовать MD5, так как исследовательские атаки предоставили достаточные основания для исключения использования алгоритма в приложениях, которым требуется устойчивость к различного рода коллизиям.

Хеши MD5 больше не считаются безопасными, и их не рекомендовано использовать для криптографической аутентификации.